数据流管理系统综述[1]

周明中　龚 俭
（东南大学计算机系华东北地区网络中心　江苏南京 210096）

摘　要：近年来对连续数据进行在线分析需求的应用不断出现，传统数据库技术已不能满足其需求，尤其是当数据来自于高速链路。由于应用的推动，各种数据分析管理系统和工具层出不穷，但其中大多数只针对个别具体应用。数据流管理系统提出了一种通用结构模型，它采用窗口机制，连续查询以及相应的优化策略迅速高效地对实时数据进行在线分析处理。本文详细地分析了数据流管理系统的框架结构和构成要素，讨论了其适用范围和优缺点，并指出当前研究中存在的问题及可能的发展趋势。

关键词：数据流；DSMS；窗口机制；连续查询；优化策略
中图分类号：TP311

Survey on Data Stream Management System

Mingzhong Zhou，Jian Gong
（Northeast Network Center, Department of Computer Science, Southeast Univ., Jiangsu, Nanjing 210096 China）

Abstract：Processing continuous dynamic data steam in real-time mode is required by many applications in the recent years. Traditional databases technologies cannot fit the needs, especially for the data from high-speed links, so that all kinds of data analysis management system and tools have emerged in, driven by applications, but most of them were simple and dedicated to special application. DSMS is a common model which adopts window mechanics, continuous query and many optimizing strategy to online analyzing and disposing real-time data quickly and efficiently. This paper comprehensively analyzes the state-of-art of structure and components of DSMS, discusses their fitness, merit and demerit, and points out the problems which exist in the recent researches and the latest possible development trends.

Key words：Data stream；DSMS；Window mechanism；Sequence query；Optimization Strategy

引言

随着计算机及其相关技术的发展，在最近几年出现了大量以数据流为信息承载模式的应用系统，譬如：工业生产，网络监测，电话通信，气象监测，经济分析等。这些数据流一般都具有实时性，连续性，顺序性以及数据量庞大等特点，使用传统的数据库管理系统已经不能满足数据流处理的要求。针对数据流管理的需求也陆续出现了一些数据查询处理系统，但这些系统要么只能提供简单的连续在线处理，不能满足相对复杂的实时处理；要么只是简单地记录流量数据，提供定期的离线查询，不能满足实时处理的要求。为满足数据流处理的需求，D. Terry, D. Goldberg et al.在[16]首先提出了应用于只增（Append-only）数据库的长期稳定运行的连续查询系统，系统只需要扫描新加入数据库的数据，故效率很高，该类系统的原型 Tapestry 最早应用于邮件和 BBS 数据的查询。各种适应于不同应用的系统紧随其后不断涌现，目前主要的数据流管理系统（DSMS）项目有 STREAM（Stanford），Aurora（Brandeis/Brown/MIT），Telegraph（Berkeley），Gigascope（AT&T），Niagara（OGI/Wisconsin），Tribeca(Bellcore)。L.Golab, M.T.Ozsu 在[1]中对现有 DSMS 进行了比较详尽的介绍，但并没有对其中所涉及的具体技术进行详细的比较分析，也没有进一步探讨目前 DSMS 中存在的问题。本文通过介绍数据流管理系统的主要内容及其最新发展情况，对其主要组成部分进行比较详细的比较和分析，并指出了 DSMS 未来可能的发展方向。

基金项目：国家自然科学基金资助项目（90104031）
作者简介：周明中（1976－），男，博士生，mzzhou@njnet.edu.cn；龚俭(联系人)，男，教授，博士生导师。

1 数据流管理系统模型

数据流管理系统（DSMS）所处理的数据流是一种实时连续的数据信息序列，而且在实际处理过程中，这种数据序列具有信息到达顺序不可控，单位时间数据到达量不均匀，数据量庞大等特点。这些特点要求 DSMS 具有应当具有以下的功能：

（1）在一定的时间内，对信息能够进行重组并处理；

（2）由于物理存储空间的限制和处理效率的要求，对数据流进行在线处理时，一般只扫描数据一遍；

（3）一般不采用阻塞的方式处理数据流，以保证数据处理的效率；

（4）能对数据进行提炼，并采取随机和/或有选择地丢包等减负措施，保证在突发流量情况下系统的整体性能；

（5）对异常的数据有足够迅速（接近于实时）的反应能力；

图 1 描述了 DSMS 的一般功能结构，用于支持长期运行，连续的，标准的，持久的查询，主要由三部分组成：输入部分，处理部分和输出部分。输入部分主要对输入的数据流进行初步的过滤，并兼有应付突发流量的基本功能(譬如在系统没有足够资源处理突发数据流时进行负载脱落)。处理部分是整个系统的主干，由三部分组成：（1）暂存子系统，数据分为三部分暂存：*静态暂存*部分存储元数据（例如 DSMS 各部分的物理位置等），允许用户自定义相关的设置；*工作暂存*部分存储当前处理的数据流，如果遇到突发流量而没有足够的物理存储空间，可以对数据进行归纳，将获得的纲要（Synposes）或摘要存储在*纲要暂存*中；（2）查询缓存，主要存储用户定义的查询条件，一般的 DSMS 系统支持用户在线修改查询条件；（3）查询处理子系统，主要功能是从查询缓存中提取用户定义的查询，根据优化需求对其进行分组，并将其作用于从输入部分获得的数据流。查询处理子系统还具有自适应功能，能根据当前数据流量和查询条件调整系统查询策略。输出部分主要的功能是保证处理所得的结果（数据流或/和关系型数据项）能够平稳地输出，一般都包含临时存储的功能。

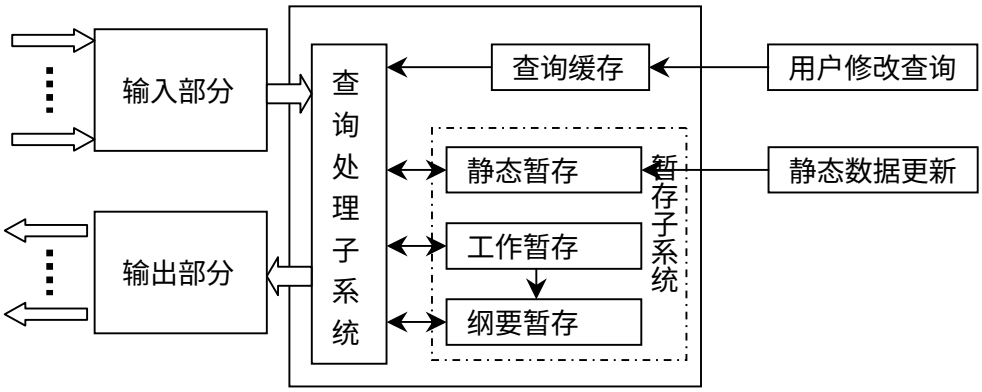

图 1.数据流管理系统的概要结构模型

2 数据流预处理模型

由于数据流只出现一次并且具有一定顺序性的特点[4], 一般在处理数据流时，DSMS 将数据流划分为若干数据单元，根据 DSMS 处理这些数据单元所采用的不同模型将数据流单元分别存储到一系列顺序的元素列表。在本节以下部分将具体介绍现有数据流管理系统对数据流查询前进行的预处理模型。

针对连续数据流的非阻塞方式处理，一般 DSMS 实验系统都采用窗口技术[2][3][5]。窗口是一种从不受限制（infinite）的数据流转化为一系列可控的受限（finite）数据单元的机制。窗口是由前后两个端点来确定的，在具体的应用中端点可以分为固定（fixed）端点，前向滑动（moving forward）

端点，后向滑动（moving backward）端点，这样就组成了九种可能的窗口形式，两端固定的窗口称为固定窗口（fixed window），两端滑动（前向或后向）的窗口称为滑动窗口（sliding window），一端固定另一端移动的窗口称为界标窗口（landmark window）。Aquery[8]，Aurora[6]，Telegraph[7]，Tribeca[9]都支持以上三种类型的窗口（其中 Telegraph 支持所有可能的窗口形式），但 STREAM[5]目前只支持滑动窗口类型。按照现有数据流管理系统使用窗口技术处理数据流的方式，可以将窗口机制分为以下三类。

- **基于顺序属性的窗口机制**

采用这种窗口机制的数据流都存在可以唯一标记流元素到达顺序的属性，现有系统一般都采用到达时间作为衡量标准[2][5][8]。这些系统通过显性或隐性的方式以一定的时间间隔给到达的数据流单元打上时间戳(timestamps)，然后根据时间戳从数据流中提取数据单元进行分析。C.Cranor, Y.Gao et al.[2]介绍的面向网络数据流监测的 Gigascope 系统。L.Golab, M.T.Ozsu [1]把这种窗口机制称为基于物理分类的窗口机制。

- **基于单元数量的窗口机制**

在这种机制下，到达的数据流被看作由一系列元组（tuple）组成队列[5][6][7]（例如在网络监测中将数据流中的每个包作为一个元组处理），应用数据流处理系统自定义的规则暂存到达的元组，并对其进行处理。Telegraph[7]将到达的元组分类存储在数据状态模块（Data SteMs），与查询状态模块（Query SteMs）进行一系列相关操作获得结果并以数据流的形式输出。

- **基于标点（punctuation）的窗口机制**

标点是确定连续流中一部分结束的记号，从而使系统可以将一条没有限制的流看成是若干受限流的组合[4]。在具体应用中，标点被作为一个控制包嵌入到输入的数据流中，数据流操作单元通过读取标点获得操作和控制数据流的信息。例如：一个标记声明数据流中某个属性 A 必须大于 10，则系统将暂存和处理属性 A≥10 的数据单元，而不对 A＜10 的数据单元进行任何操作，其功能相当于对连续的数据流以属性 A 的值等于 10 为分界进行分组，直到另一个声明相关属性的标记到达。使用标记技术的数据流处理项目主要包括 Aurora[6]，Niagara[10]，STREAM[5]，Hancock[11]，Tribeca[9]等。

在目前 DSMS 系统中，这几种窗口机制是一般都是结合使用的，已达到不同阶段采用相对更有效处理方式的目的，如 STREAM 中就包含这三种类型的窗口机制。

3 数据流查询过程

数据流查询操作和一般的 RDBMS 提供的查询有很多相似的地方，但是由于数据流本身的特点，数据不再像传统的存储在数据库中的数据一样通过“拉”（pull）的方式获得，而是通过“推”（push）的方式提供给查询系统。这要求数据流查询机制有更高的处理效率、更好的自适应性以及处理突发流量进行负载平衡的能力。

3.1 连续查询理论基础

所谓连续查询，其语义是查询结果为这个查询在此前任意时刻产生数据的集合。连续查询的语义是时间独立的，也就是说不同用户使用同一个连续查询将得到相同的结果[2]。

假设每一个时钟周期进行一次查询操作，Q(t)表示在时间 t 输出的数据集合。当一个查询是连续查询时，它在 t 时刻总输出的集合 $Q_M(t)$可以用公式（1）表示：

$$Q_M(t) = \bigcup_{S=0}^{t-1} Q(s) \qquad (1)$$

D. Terry, D. Goldberg, et al.在[16]通过推导证明了在单调情况下，公式（1）可以写成以下形式，这里

所谓的单调模式是指在 $t_1 < t_2$ 的条件下 ，必然有 $Q_M(t_1) \subset Q_M(t_2)$。

$$Q_M(t) = \bigcup_{i=1}^{t-1}(Q_M(i) - Q_M(i-1)) \bigcup Q_M(0) \qquad (2)$$

从公式（2）可以看出，在单调模式下，连续查询只需要计算新到的数据单元，将所得新的元组子集添加到结果集合中，这样在很大程度上节省了计算资源。如果为非单调模式，即查询结果集中的元组可能随时间变化而发生状态变化，则每次计算都需要从头开始。当连续查询中存在＝，≠，<，> 这些比较算子时，一般可以推出此查询为非单调的结论[16]。

A. Arasu. et al.在[12]中对数据流中连续查询的单调模式和非单调模式都作了比较深入的探讨。

3.2 数据流查询语言

在 DSMS 中，连续查询可以被分解为算子（operator）的集合，例如 project，select，join 等等，这些算子在传统的 DBMS 中也作了相关定义，其区别主要集中在 DSMS 如何使算子能够在非阻塞的模式下运行，这个问题可以使用窗口技术解决（上一节已经介绍）。

数据流查询操作的整个过程可以使用单向无环图（DAG）来表示，其中每个点表示一个管道（pipelined）算子，有向的边表示连接两个算子的队列（例如，经过算子 A 处理的数据流通过缓冲写入队列 AB 成为算子 B 的输入流）。

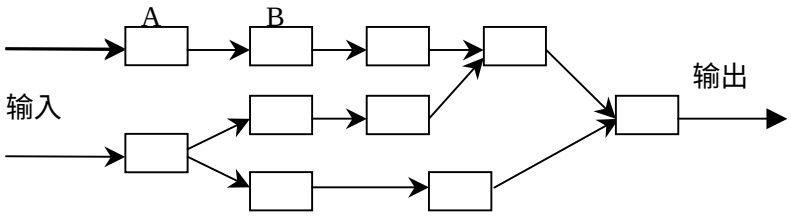

图 2

虽然在数据处理模型上所有 DSMS 系统都比较相似，但具体的实现系统会根据应用的不同而采用了不同的机制[2][5][6][7][8][9][10]。这些机制具体可以归纳为一下几种类型。

- **基于关系的机制**

基于关系的机制主要是指通过流－关系算子（operator）将连续的数据流转化为关系模型进行处理，得到关系型结果可以直接输出，或通过关系－流算子转化为数据流输出。AQuery[21]，CQL[12][22]，StreaQual[7][23][24]都是基于关系模型的连续查询语言。

AQuery 是一种实现顺序依赖的类 SQL 语言，所谓顺序依赖是指如果输入数据的顺序不一样其结果也不一样（例如，网络流的到达是时间相关的）。Aquery 通过 ORDER BY 语句将数据流作为一系列排列（array）基于顺序按行存储在表中，这样就可以通过基于顺序的方式定义操作子，如：first，next，previous，last 等。以下是 Aquery 针对一组网络数据包进行查询的例子，目的是将一定时间范围内的包按照源宿地址和时戳超过 120 的类型进行分组计算，获得每一个组的源宿地址，包的平均长度和时戳的数目（也就是包的数目）。

```
SELECT src，dest， avg（length），count（timestamp）
FROM Packets
        ASSUMING ORDER src，dest，timestamp
GROUP BY src, dest, sums(deltas(timestamp)>120)
```

StreaQual 也是利用窗口技术将连续的数据流分解为固定长度的分片，然后利用关系模型对这些数据流分片处理，同时还提供给用户自定义分片长度的函数接口。

CQL 是 STREAM 系统中采用的查询语言，它通过指定数据流的某种属性范围来定义窗口的大

小。下面的例子是在网上拍卖中，计算最近一个小时编号在 100－200 之间的拍卖品的活动次数：

SELECT Count（*） From Bid[Range 1 Hour]

Where item_id >= 100 ans item_id <= 200

CQL 也支持无窗口的连续查询，其前提条件是查询不需要历史数据。

- **基于对象的机制**

从数据流中提取相关的元素按层次类型处理是基于对象机制的查询语言的最主要的特点，譬如在 Tribeca 的数据描述语言（DDL）就将网络中采集到的数据包头作为一个对象处理，UDP/IP 头和 TCP/IP 头的继承于 IP 头，而各种应用层的头又是分别继承于 TCP/IP 头和 UDP/IP 头，等等。由于基于对象的机制采用了层次结构，数据流管理系统设计采用的查询框架也是基于层次结构的，对数据结构比较复杂的数据流，这种查询框架有较大的优势：可以将任务有效地分解，前期对大量数据流进行简单分类，后期对相对少量的数据流进行具体处理，从而达到总体性能的均衡。

- **基于过程的机制**

基于过程的查询语言提供了一种用户自定义查询方案的机制，用户可以通过组织算子组合出查询框架，从而控制数据流的流向并尽可能获得更多的信息。在 Aurora 系统中，提供了一个基于 Java 的图形用户接口，用户可以定义并随时修改查询计划。其查询计划图和图 2 相类似，是由一系列代表算子的点和代表数据流向的有向弧组成[6]。

所有的数据流处理都是通过窗口机制来实现对一段时间的历史数据进行回溯比较和总结，如果无需对历史记录进行进一步处理则都可以支持无窗口限制的查询。基于关系的查询语言应用比较普遍，能比较简洁地刻画数据流的特点，具体操作也比较简单，一般适用于数据结构相对简单，查询结构相对稳定的数据流查询；基于对象的机制可以应用于比较复杂的数据结构的数据流查询，通过层次对象模型简化查询的结构；针对查询结构需要动态调整的数据流，采用基于过程的机制是较好的选择，因为前两种机制查询结构相对固定，不适合频繁调整。

4 数据流管理系统优化策略

数据流处理过程中需要解决的问题很多，其中比较典型的有两个：突发流量不确定性和近乎实时处理的性能需求。这些问题和系统有限的软硬件资源形成了冲突，需要采取必要的措施进行协调，以达到在一定条件下的处理性能最优化。本节的其余部分将具体介绍目前在 DSMS 实现项目中比较流行的几种系统性能优化策略。

4.1 进度安排（Scheduling）

进度安排是一种积极的优化策略，是影响系统的整体性能最关键的因素之一。理想的进度安排策略应当能够保证：（1）在资源一定的情况下获得最优的性能；（2）及时发现系统过载，并触发相应解决机制（如突发流量处理）；（3）保证特定查询的 QoS 需求；（4）部署简单，操作方便[15]。在部署进度安排策略时，主要性能指标包括：对元组的响应时间，内存需要量，吞吐量，精确度等。在实施过程中，系统为达到各种性能指标的要求会发生冲突（例如较小的相应时间必然要求较大的内存），所以对实际系统而言，寻找各种指标的平衡至关重要。具体的 DSMS 会根据数据流的特点和要达到的目的，以部分牺牲其他性能为代价，选取其中的一个或一部分性能作为主要衡量指标。

B.Babcock, M. Datar, et al. 在[14]提出了应用于 STREAM 项目的 Chain 策略，其主要任务是在系统进行任何查询操作时，最小化所占物理存储空间；Aurora 提供了基于输出响应时间的 QoS 方案[6]；Q.Jiang, S.Chakravarthy 在[15]提出了一种基于算子路径的进度安排策略 Path capcity scheduling（PCS），通过为具有最大能力处理的路径安排进度的方式来达到最小化响应时间的目的，并在其基础上，综合 Chain 策略的优点，提出了 Segment Scheduling 策略，通过将算子路径分解为若干片断，在处理占用空间控制方面取得了较好的效果。

Chain 策略虽然通过对算子运算次序的调整，有效的解决了内存占用问题，但这是建立在系统有足够物理资源的假设基础上，并且只处理单条输入数据流的查询，当突发流量维持较长时间时，系统的响应时间会延长并有可能产生饥饿的现象[14]。虽然 PCS 在运行过程中采用了基于输入流大小一致的假设的内存优化方法，但是仍然在内存占用方面的性能要比 Chain 策略差。Segment Scheduling 策略虽然在一定周期可以比 Chain 策略释放更多的资源并有较好的响应时间，但在具体运行时所占用的空间要大于 Chain 策略。

从[15]给出的试验结果看，在响应时间和占用内存方面，三种进度安排算法各有所长，但总体而言，Chain 策略的吞吐量要略低于其他两种算法。

4.2 负载脱落（Loadshedding）

当突发流量超过系统的处理能力，如果不采取相应的措施，会导致整个系统的吞吐量和响应时间都恶化。负载脱落通过丢弃一定数量的数据，在部分牺牲准确性和完整性的条件下，保证系统的性能。负载脱落算法可以根据采用的处理方式分为以下两种：随机的负载脱落算法和基于语义的负载脱落算法。

随机负载脱落是指在发现数据输入超出系统处理能力时，通过按一定的比重随机丢弃部分元组保证系统的正常运行。基于语义的负载脱落，通过用户对流处理语义的理解，有选择地丢弃一部分元组，使元组损失对系统性能和输出结果的影响最小化。

随机负载脱落的算法比较简单，所需占用的内存和计算量都比较小，但是由于没有考虑选择丢弃元组对总体性能和输出的影响，在某些情况下可能导致系统性能的恶化和输出结果的精确性，所以目前普遍采用的负载脱落算法一般是基于语义的[17][22][18][19][25]。基于语义的负载脱落算法与系统的上下文有关，主要考虑的问题是：何时，何地以及如何进行。

D.Carney et al.在[25]中提出了通过丢弃元组实现的随机负载脱落和通过过滤元组实现的语义负载脱落，过滤指有控制地丢弃一些不重要的元组来保证系统的 QoS。B. Babcock, M. Datar,et al.在[19]中指出了 D.Carney et al.的不足——该基于语义的负载脱落并不能有效地保证查询的精确性，从而提出改善精确性的方法，并通过配置随机抽样算子来具体实现基于语义的负载脱落。

4.3 近似值（Approximate）计算

当系统性能满足数据流处理的要求时，处理结果可以正常输出。但是如果处理要求超出系统的处理能力时，如突发流量、处理数据流所需内存超出物理内存等，可能导致系统总体性能下降甚至出现错误。DSMS 采用计算近似值的方法，通过发现处理数据流的相似性，将其进行归类处理，在部分牺牲精确性的条件下获得处理时间的减少和存储空间上的节省。

近似值计算是 DSMS 中重要的数据处理方式，所采用的算法根据具体方法的不同分为： 计数(Counting)，哈希（Hashing），抽样（Sampling），概要（Sketches），直方图（Histograms）和小波（Wavelets）等。

计数算法通过计算分位数，统计经常出现的单元集合，用存储所选择类型单元的出现频度的方法减少存储空间，E.Demaine et al 在[30]中阐述了利用计数算法统计 Internet 高速路由中出现频度最高的 k 种类型的包。哈希算法将不同的元素按照独立的哈希函数映射到 k 个桶中，通过统计桶的大小揭示类型的出现频繁程度。抽样算法按一定的比率选取样本以替代维护所有元素的信息，对样本比率的提高可以有效地提高算法的精确度。概要算法最初是 N.Alon, Y. Matias,在[26]首先提出来的，并在随后几年对该篇论文改进，其主要思想是通过计算数据集内元素之间“距离”的方式，用有限的空间存储数据流的摘要信息。直方图方法是用于分析数据集内数据特性分布情况的常用方式，B.Babcock et al 在[3]中提出了应用于数据流摘录的三种直方图模型：V-Optimal Histogram，Equi-Width Histograms，End-Biased Histograms。小波变换算法是利用小部分系数的集合替代潜在的信息，近期在数据流近似值计算中利用比较广泛。J. Vitter, M.Wang et al.在[37][38]分别利用小波变换，计算数

据立方近似值、直方图和多维聚集近似值。

目前计算近似值的方法有多项算法相结合，取长补短的趋势，M.Charika, K.Chen et al.在[41]中利用计数和概要算法结合形成了新的算法：Count Sketch，[35]介绍了基于小波的直方图方法等。

5 未来研究方向

虽然数据流管理系统在不同领域数据流处理中得到了广泛的应用，但是由于其在总体上尚处于起步阶段，而数据流处理本身需求有多种形式且复杂程度呈迅速上升趋势，所以数据流管理系统在许多方面有待提高，主要集中在以下：

1、数据流管理系统的统一接口问题

目前相关研究机构出于各自研究的需要，都自定义了一系列自己的 DSMS 接口规范，而没有考虑不同 DSMS 之间的互连和共享。由于所需处理问题的复杂化和各个研究机构之间协作的增加，和传统的 DBMS 类似，定义统一访问接口势在必行。

2、数据流处理优化策略

由于所需处理数据的量和数据处理的复杂程度增长的速度要远远超过硬件处理能力的增长速度，而且现有数据流处理优化策略还不能很好地满足相关要求，所以有必要进行进一步改进和发展流处理优化策略。首先针对进度安排，问题主要集中在响应时间和内存占用之间能否找到一个平衡点，并且平衡点是可变的，始终契合具体的应用需求；负载脱落算法主要考虑如何跟语义更紧密地结合，从而使其代价最小化；近似值计算目前主要发展方向是多种处理方法的结合使用，具体方法的选择应当较好地考虑其使用范围和具体应用。

3、数据流管理系统如何在整体性能方面超过针对应用的专业工具

数据流管理系统相对于一般针对具体应用的专业工具主要优点在于其适应能力和通用性，用户可以根据具体的需要通过定义不同的接口来获得不同的测试结果，但这是以牺牲一定的处理性能为代价的。如何通过技术的改进在整体性能方面超过一般专用的工具；或者通过为一般工具提供接口的方式和专用工具结合使用，以达到更高的处理能力和更好的处理效果，这是 DSMS 发展所需解决的一个问题。

数据流管理系统在借鉴传统数据库的基础上提出了时间戳，窗口机制，连续查询，优化策略等方法，用于在线分析连续数据流。在现有 DSMS 中这些技术得到了广泛地应用和发展，但是由于硬件性能增长速度要远远落后于数据流处理需求的增长速度，所以有必要在系统结构和优化算法等诸多方面促进 DSMS 的发展。

参考文献

[1]L.Golab,M.T.Ozsu. Issues in Data Stream Management-A Survey. In *SIZGMOD Record*, 32(2), 2003.

[2]C.Cranor,Y.Gao, et al. GigaScope:High Performance Network Monitoring with an SQL Interface. In *Proc. ACM Int. on Management of Data*, pages:623, 2003.

[3] B.Babcock, M. Datar, R. Motwani, J.Widom. Models and Issues in Data Stream Systems. In *PODS*, 2002

[4]M.Henzinger,P.Raghavan et al. Computing on Data Streams. In *Series in Discrete Mathematics and Theoretical Computer Science*, vol 50, 1999

[5]The STREAM Group. STREAM：The Stanford Stream Data Manager. Stanford University. http://www-db.stanford.edu/stream., Mar, 2004.

[6]D.J.Abadi, D.Carney, et al. Aurora: A Data Stream Management System. In *ACM SIGMOD Conference*, Page 666, San Diego, CA, June 2003.

[7] S. Chandrasekaran, O. Cooper, A.Deshpande, et al. TelegraphCQ: Continuous Dataflow Processing for an Uncertain World. In *CIDR*, 2003.

[8] Y.Zhu, D.Shasha. StatStream:Statistical Monitoring of Thousands of Data Streams in Real Time. In *Proc. 28th Int. Conf. On Very Large Data Bases.* pages: 358-369, 2002.

[9] M.Sullivan, A. Heybey. Tribeca: A System for Managing Large Databases of Network Traffic.In *Proceedings of the 1998 USENIX Annual Techincal Conference*. New Orleans, LA, June 1998.

[10]J.Chen, D.DwWitt, F.Tian, Y.Wang. NiagaraCQ: A Scalable Coutihuous Query System for Internet Databases. In *Proc. ACM Int. Conf. on Management of Data,* pages: 379-390,2000.

[11]D.Banchea, K.Fisher, et al. Hancock: A Language for Processing Very Large-scale Data. In *USENIX 2nd Conf. on Domain-Specific Language.* pages:163-176, Oct. 1999.

[12]A. Arasu, S.Babu, J. Widom. A Abstract Sematics and Concrete Language for Continuous Queries over Streams and Relations. Stanford University Technical Report 2002-57, Nov. 2002. Available at http://dbpubs.stanford.edu:8090/pub/2002-57, Mar,2004.

[13]S.Madden, M.J. Franklin. Fjording the Stream: an Architecture for Queries Over Streaming Sensor Data. In *Proc. 18th Int. Conf.on Data Engineering*, pages.555-566, 2002.

[14]B.Babcock, S.Babu, et al. Chain: Operator Scheduling for Memory Minimization in Data Systems. In *Proc. ACM Int. Conf. On Management of Data*, June 2003.

[15]Q.Jiang, S.Chakravarthy. Scheduling Strategies for a Data Stream Management System. From *itlab.uta.edu/cse6392/Fall2003/Selected Papers/ DUSTI 20/dsms-scheduling.pdf*, Mar,2004.

[16] D. Terry, D. Goldberg, D. Nichols et al. Continuous Queries over Append-Only Databases. In *SIGMOD Conference*. pages:321-330, 1992.

[17] A.Das, J.Gehrke, et al. Approximate Join Processing Over Data Streams. In *Proc. of the 2003 ACM SIGMOD Int. conf. on Management of data*. pages. 40-51, 2003.

[18]N.Tatbul, U.Cetintemel, et al. Load Shedding in a Data Stream Manager. In *Proc. of VLDB*, Berlin, Germany, 2003.

[19] B.Babcock, M.Datar, et al. Load Shedding Techniques for Data Stream Systems. In *Management and Processing of Data Streams*, San Diego, California, USA. Jun. 2003.

[20] C.Cortes, K.Fisher, et al. Hancock: A Language for Extracting Signatures from Data Streams. In *Knowledge Discovery and Data Mining*. pages.9-11, 2000.

[21] A.Lerner, D.Shasha. Aquery: Query Language for Ordered Data, Optimization Techniques, and Experiments. TR2003-836, Courant Institute of Mathematical Sciences, New York University, Mar. 2003, Available at *http://csdocs.cs.nyu.edu/Dienst/UI/2.0/Describe/ncstrl.nyu_cs%2FTR 2003-836.*

[22]R.Motwani, J.Widom, et al. Query Processing, Approxination, and Resource Management in Data Stream Management System. In *Proc. 1st on Innovative Data System*. 2003, pp.245-256.

[23]S.Chadrasekaran, M.J.Franklin.. Streaming Queries over Streaming Data. In *Proc. International Conference on Very Large Data Bases (VLDB),* HongKong, Aug. 2002.

[24] S.Chadrasekaran, M.J.Franklin.. PSoup: a system for streaming queries over streaming data. In *VLDB Journal*, Aug. 2003.

[25]D. Carney,U. Cetintemel et al. Monitoring streams – a new class of data management applications. In *Proc. 28th ACM Symp. On Theory of Computing*, pages.20-29, 1996.

[26] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *JCSS: Journal of Computer and System Sciences*: 58, 1999.

[27] S. Guha and N. Koudas. Approximating a data stream for querying and estimation: Algorithms and performance evaluation. In *IEEE International Conference on Data Engineering*, pages 567--576, 2002.

[28] M. Ajtai, T. S. Jayram, et al. Approximate counting of inversions in a data stream. *STOC: Symposium on the Theory of Computing* . pages: 370-379,2002.

[29] Y. Zhu, D. Shasha: Efficient elastic burst detection in data streams. In *Intl. Conf. on Knowledge Discovery and data mining*. pages: 336-345, 2003.
[30] E.Demaine, A.Lopez-Ortiz et al. Frequency Estimation of Internet Packet Streams with Limited Space. In *Proc. Europeam Symposium on Algorithms*. pages 348-360, 2002.
[31] P.B. Gibbons, Y.Matias. Synopsis Data Structures for Massive Data Sets. In *Proc. 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 909-910, 1999.
[32] K.Chakrabarti, M.N.Garafalakis et al. Approximate query processing using Wavelets. In *Proc. Of the 2000 Intl. Conf. on Very Large Data Bases*. pages 111-122, Sept, 2000.
[33] A.Gilbert, S.Guha at al. Fast, small-space algorithms for approximate histogram maintenance. In *Proc. Of the 2002 Annual ACM Symp. on Theory of Computing.* 2002.
[34] Y.E.Ioannidis, V.Poosala. Histogram-based approximation of set-valued query-answers. In *Proc. Of the 2000 Annual IEEE Symp. On Foundations of Computer Science*. pages 189-197, 2000.
[35] Y.Matias, J.Vitter et al. Wavelet-Based Histograms for Selectivity Estimation. In *Proc. of the 1998 ACM SIGMOD Intl. Conf. on Management of Data*. pages: 448-459, June 1998.
[36] Y.Matias, J.Vitter et al. Dynamic maintenance of Wavelet-based Histograms. In *Proc. Of the 2000 Intl. Conf. on Very Large Data Bases*. pages 102-110, Sept., 2000.
[37] J. Vitter, M.Wang et al. Data cube Approximation and Histograms via wavelets. In *Proc. Of the 1998 Intl. Conf. on Information and Knowledge Management*. Nov. 1998.
[38] J. Vitter, M.Wang et al. Approximate computation of multidimensional aggregates of space data using Wavelets. In *Proc. Of the 1999 ACM SIGMOD Intl. Conf. on Management of Da*ta, pages 193-204, June, 1999.
[39]J.Naughton, D.DeWitt, et al. The Niagara Internet Query System. Available at http://www.cs.wisc.edu/niagara/papers/NIAGRAVLDB00.v4.pdf , 2000.
[40] M.Charikar, K.Chen et al. Finding frequent Items in data streams. In *Proc. 29th Int. Colloquium on Automata, Languages and Programming*. pages:693-703, 2002.
[41] D.Carney, U.Cetintemel，et al. Operator Scheduling in a Data Stream Manager. In *Proc. Of the 29th Int. Conf. on VLDB*, Sept. 2003.
[42]B.Bavcock, C.Olston. Distributed Top-k Monitoring. In *Proc. 2003 ACMSIGMOD International Conf. on Management of Data*. 2003, San Diego, California.